

Introduction

- **Efficient application development and migration for FPGA-based systems - *YES, IT EXISTS!***
 - Compilers for software code to hardware circuits
 - Tools for kernel connectivity to system interfaces
 - Scripting for synthesis and place-and-route (PAR)
 - Multi-user runtime management system
- **SRC Computers, LLC provides:**
 - Carte™ Programming Environment with ANSI C and Fortran
 - System firmware with compiler integration
 - ‘GNU make’ build process with fully integrated toolflow
 - Robust runtime system for resource management

Features & Challenges for FPGA Tools

- **Application Migration**
 - Must effectively represent spatial and temporal parallelism
 - Minimize deviation from original application source code
- **System Integration**
 - Efficiently scope application to available FPGA resources
 - Manage memory bandwidth and accessibility
 - Manage heterogeneous peer-processor interaction
- **Runtime Libraries**
 - Dynamically assign system resources based on system load
 - Manage resources effectively within the chosen architecture
- **Encapsulation of Build Environments**
 - Functional emulation, circuit simulation, and hardware compilation
 - Simple interface with GNU make and Unix-style execution



SRC-7 ATLAS
Multi-Module System

SRC® Toolflow

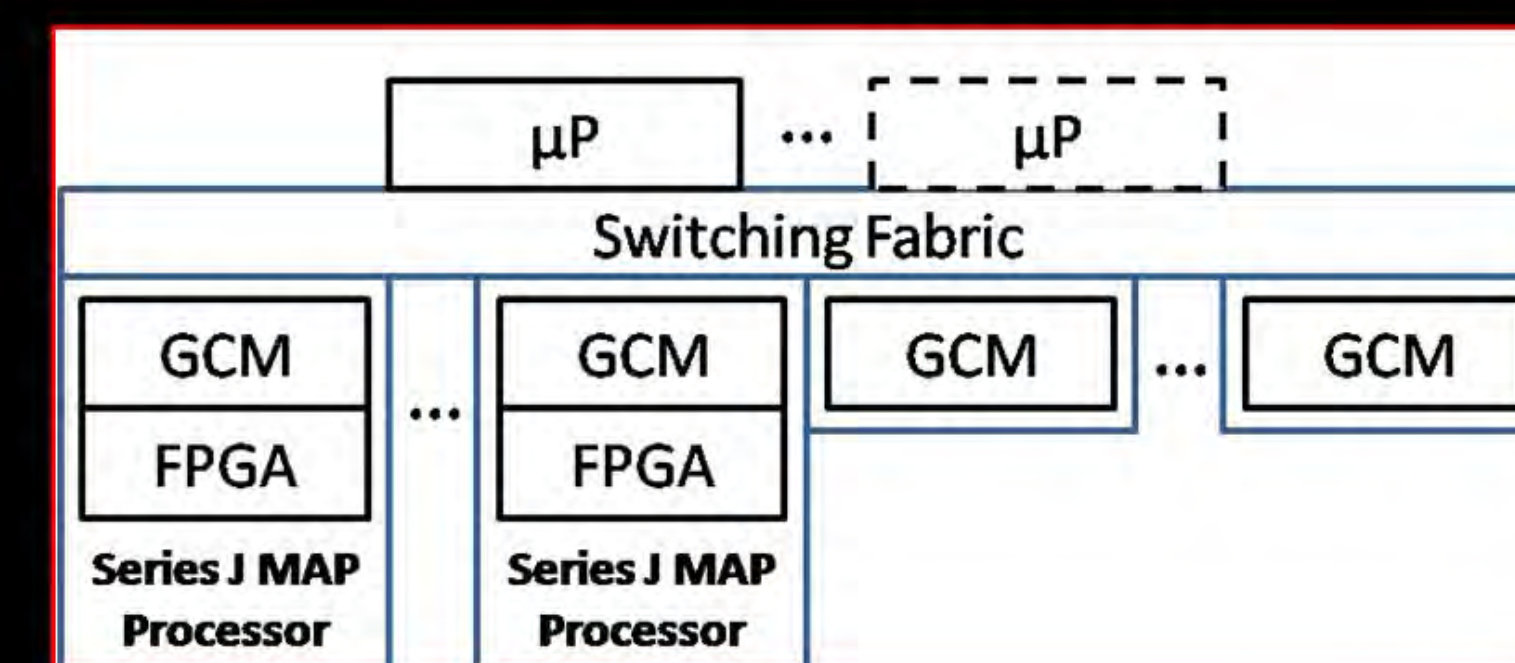
- **Application Migration**
 - Carte™ toolflow with ANSI C and Fortran development and migration of application codes for MAP® processors
- **System Integration**
 - Macro-based library approach allows usage of key architectural features while hiding implementation details
- **Runtime Libraries**
 - MAP processors and memories are requested from the manager and regulate multi-user traffic and behavior
- **Encapsulation of Build Environments**
 - Functional emulation for debugging, also circuit simulation
 - GNU make interface and generation of unified executables

The Tools Have Arrived

Two-Command Compilation and Execution of Scalable Multi-FPGA Applications

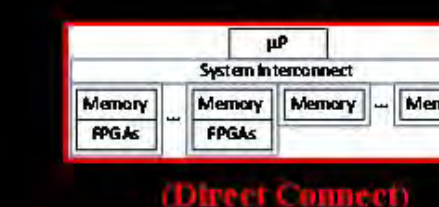
Brian Holland
SRC Computers, LLC

SRC-7 System Architecture



Individually addressable Series J MAP® Processors and Global Common Memories (GCM)

Based on direct connect architecture

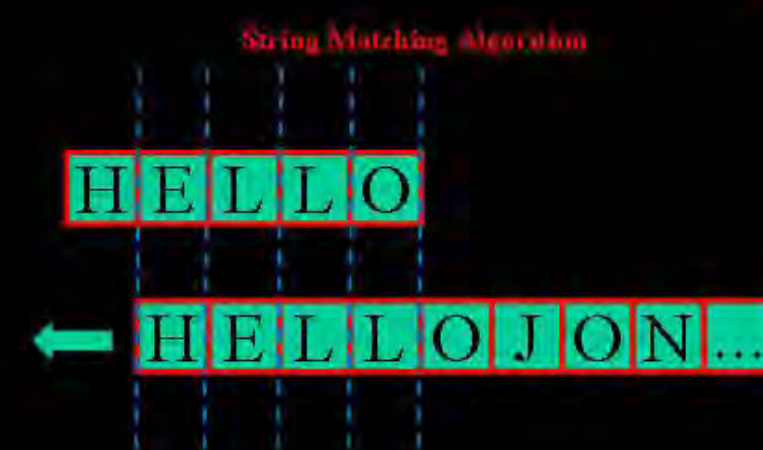


SRC-7 Mid-Sized Rugged Multi-Module System



Case Study: String Matching

- **Case Study**
 - Illustrates efficient migration of legacy codes to SRC® Carte™ Programming Environment
 - Computation, memory management, and parallelism handled with application code
- **Carte String Matching**
 - Pipelined comparison of keywords with larger database
 - Useful in fields such as packet inspection, data mining, and computational biology



```
Legacy String Matching Algorithm
1: uint64_t data, result, keyword, mask;
2: char *mem = (char *)malloc(numchars);
3: char *res = (char *)malloc(numchars);
4: uint64_t data, result, keyword, mask;
5: keyword = 0x48454C4C4F //ascii HELLO
6: mask = 0x000000FFFFFFFF; //5 char window
7:
8: for(i=0; i<numchars; i++) {
9:   data = (data<<8) | mem[i];
10:  res[i] = (data & mask) == keyword;
11: }
```

Two-Step Compilation and Execution

```
src7> make
src7> ./string_match
```

Quantitative Results

- **Toolflow Performance**
 - Emulation requires significantly less time than FPGA hardware compilation
 - Hardware execution is extremely rapid
 - Performance approaches 1000x improvement over microprocessor
- **Efficiency**
 - Overhead latency becomes negligible for sufficiently large data sets

Table I: Toolflow Performance – 1MB Text

	Compilation Time (s)	Execution Time (s)
Software Emulation	2.6E+0	1.0E+2
Physical Hardware	4.2E+3	3.2E-1

Table II: Streaming and Pipelining Efficiency

Size (B)	Clock Cycles	Efficiency (%)
1K	1396	73.3
10K	10617	96.4
100K	102772	99.6
1M	1048953	99.9+



SRC-7 Rack Systems